CA Signed Certificates configureren met IOS XE PKI

Inhoud

Inleiding

Dit document dient als algemene handleiding voor het configureren van IOS XE-certificaten die zijn ondertekend door een certificeringsinstantie van een derde partij (CA).

In dit document wordt beschreven hoe een meerlagige CA Signed-keten kan worden geïmporteerd zoals voor het apparaat om als identiteitsbewijs te fungeren, en hoe andere certificaten van derden kunnen worden geïmporteerd met het oog op de validering van certificaten.

Voorwaarden

Vereisten

NTP- en kloktijd **MOETEN** worden geconfigureerd bij gebruik van IOS PKI-functies.

Als een beheerder NTP niet configureert, kunt u problemen hebben met een certificaat dat wordt gegenereerd met een toekomstige/afgelopen datum/tijd. Deze scheefheid in datum of tijd kan importproblemen en andere problemen in de toekomst veroorzaken.

NTP-voorbeeldconfiguratie:

```
ntp server 192.168.1.1
clock timezone EST -5
clock summer-time EDT recurring
```

Gebruikte componenten

- Cisco router met Cisco IOS® XE17.11.1a

De informatie in dit document is gebaseerd op de apparaten in een specifieke laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u zorgen dat u de potentiële impact van elke opdracht begrijpt.

Achtergrondinformatie

Sommige functies die in dit document zijn beschreven, zijn mogelijk niet beschikbaar in oudere IOS XE-versies. Waar mogelijk is het document gedocumenteerd wanneer een opdracht of een functie is ingevoerd of gewijzigd.

Raadpleeg altijd de officiële documentatie voor IOS XE PKI-functies voor een bepaalde versie om eventuele beperkingen of wijzigingen te begrijpen die mogelijk relevant zijn voor uw specifieke versie:

Voorbeelden:

- IOS 15 M/T: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_conn_pki/configuration/15-mt/sec-pki-15-mt-book/sec-pki-overview.html
- IOS XE 16.12.x: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_conn_pki/configuration/xe-16-12/sec-pki-xe-16-12-book/sec-est-client-supp-pki.html
- IOS XE 17.x: https://www.cisco.com/c/en/us/td/docs/routers/ios/config/17-x/sec-vpn/b-security-vpn/m_sec-pki-overview-0.html

IOS XE PKI-configuratie

Op hoog niveau moet een beheerder de volgende acties uitvoeren wanneer hij met IOS XE PKI-certificaten werkt:

1. Maak een sleutel voor gebruik met een functie of service (**crypto-sleutel gegenereerd**)
2. Configureer een trustpoint met verschillende parameters en koppel de toets aan elkaar. (**crypto pki trustpoint**)
3. Genereren van een certificaat ondertekeningsaanvraag (CSR) (**crypto pki enroll**)
4. Verstrek de MVO aan een CA voor ondertekening (*niet gedekt in dit document*)
5. Verifiëren van de Root en/of tussentijdse CA-certificaten (**crypto pki authenticate**)
6. De apparaatcertificaten importeren (**crypto pki import**)
7. Optioneel: Verifiëren peer CA certificaten (**crypto pki authenticate**)

Deze stappen zijn beschreven in de komende secties die gegroepeerd zijn op de opdrachten die voor de gegeven handeling vereist zijn.

crypto-sleutelgeneratie

Veel beheerders zijn dit commando ingegaan om Secure Socket Shell (SSH) in te schakelen op een router of als onderdeel van een configuratiehandleiding voor een functie. Maar weinigen hebben niet ontleed wat de opdracht eigenlijk doet.

Neem bijvoorbeeld de onderstaande opdrachten:

```
crypto key generate rsa general-keys modulus 2048 label rsaKey exportable
crypto key generate ec keysize 521 exportable label ecKey
```

Als u deze opdrachten in de specifieke onderdelen scheidt, wordt het gebruik gedetailleerd:

- Het eerste deel van de opdracht in zwart (crypto key generated) instrueert de router dat wij een nieuwe sleutel zullen creëren. Er zijn andere opties zoals crypto key export, crypto key import, of crypto key zerosize die later zullen worden gedetailleerd.
- Het volgende deel van de opdracht in **groen** (rsa general-keys, ec) instrueert de router precies welk type sleutel we maken. Voor de meeste doeleinden zal een Rivest-Shamir-Adleman (RSA) sleutelpaar bestaande uit een publieke/private sleutel worden gebruikt, maar een beheerder kan ook een elliptische curve (EC) configureren voor gebruik met functies zoals die waarvoor ECDSA-certificaten nodig zijn of voor gebruik met ECDHE handshakes.
- De opdracht in **oranje** definieert de grootte van onze sleutel.
    - Voor RSA is de modulus de terminologie en de waarden zoals tussen 512-4096 zijn beschikbare opties. De standaardmodulusgrootte varieert per versie, maar het wordt voorgesteld om de beste praktijken van Cisco voor volgende generatiecryptografie te volgen en sleutels te gebruiken groter dan 2048.
    - Voor EC is de opdracht sleutelgrootte vereist om het aantal bits in de sleutel te specificeren. De opties zijn 256, 384 of 512.
- De opdracht in **paars** definieert het label voor deze toets. Dit is belangrijk omdat een beheerder mogelijk meerdere sleutels voor verschillende doeleinden op hetzelfde IOS XE-apparaat moet definiëren. Het etiket wordt gebruikt om de nauwkeurige sleutel voor gebruik met een bepaalde eigenschap te specificeren. Gebruik waar mogelijk altijd een label om de sleutels in gebruik te onderscheiden en het toewijzen van sleutels aan functies veel gemakkelijker te maken. Bijvoorbeeld: label SSH, label CUBE, label HTTPS zal twee sleutels creëren voor gebruik met verschillende diensten of functies.
    - Het standaardlabel voor een sleutel is de apparaten hostname.domain. Sommige apparaten kunnen RSA-toetsen genereren op de eerste boot. Door geen label in te voeren na de correctie, kan een beheerder het risico lopen om per ongeluk de verkeerde toets te overschrijven of te regenereren
- De laatste opdracht in **blauw** is de uitvoerbare postfix. Deze opdracht geeft aan dat de sleutel kan worden gebruikt met de **crypto pki export** commando voor export en gebruik met andere systemen. Een voorbeeld kan zijn om te importeren in een peer High Availability-apparaat, zodat een enkele sleutel wordt gebruikt door beide leden van een HA-paar of voor gebruik binnen probleemoplossing tools zoals Wireshark om RSA-gebaseerde TLS-sessies te decrypteren. Wat ook de reden moet worden vermeld dat de RSA-sleutels alleen kunnen worden gemaakt als exporteerbaar vanaf het begin. Als een beheerder een niet-exporteerbare RSA-sleutel maakt, kan deze sleutel niet worden ingesteld als exporteerbaar zonder de sleutel te regenereren, wat gevolgen kan hebben voor andere functies, zoals het ongeldig maken van alle certificaten die met die sleutel zijn gemaakt. Dat gezegd zijnde, kan een exporteerbare sleutel worden gedegradeerd naar niet-exporteerbaar zonder de sleutel te regenereren door de opdracht **crypto sleutel verplaatsen rsaKeyLabel niet-exporteerbaar**

Configuratievoorbeelden:

<#root>

```
Router(config)#
```

```
crypto key generate rsa general-keys modulus 2048 label rsaKey exportable
```

```
The name for the keys will be: rsaKey

% The key modulus size is 2048 bits
% Generating 2048 bit RSA keys, keys will be exportable...
[OK] (elapsed time was 1 seconds)

Router(config)#
```

```
crypto key generate ec keysize 521 exportable label ecKey
```

```
The name for the keys will be: ecKey
```

Controlevoorbeelden:

<#root>

```
Router#
```

```
show crypto key mypubkey rsa rsaKey
```

```
% Key pair was generated at: 10:21:42 EDT Apr 14 2023
Key name: rsaKey
Key type: RSA KEYS      2048 bits
 Storage Device: not specified
 Usage: General Purpose Key
 Key is exportable. Redundancy enabled.
 Key Data:
  30820122 300D0609 2A864886 F70D0101 01050003 82010F00 3082010A 02820101
  [..truncated..]
  9F020301 0001

Router#
```

```
show crypto key mypubkey ec ecKey
```

```
% Key pair was generated at: 10:03:05 EDT Apr 14 2023
Key name: ecKey
Key type: EC KEYS      p521 curve
 Storage Device: private-config
 Usage: Signature Key
 Key is exportable. Redundancy enabled.
 Key Data:
  30819B30 1006072A 8648CE3D 02010605 2B810400 23038186 000401A2 A77FCD34
  [..truncated..]
  93FAC967 96ADA79E 4A245881 B2AD2F4A 279A362D F390A20F C06D5845 06DA
```

crypto pki trustpoint

Trustpoints zijn een 'mapachtig' concept voor het opslaan en beheren van PKI-certificaten binnen IOS XE. (Command Syntax)

Op hoog niveau:

1. Elk IOS XE-trustpoint kan een enkele hoofdmap of een tussenliggend CA-certificaat bevatten dat is geladen via de **crypto pki authenticate** opdracht. Denk aan geverifieerde trustpoints als het toevoegen van certificaten die nu worden vertrouwd door het apparaat.
2. Elk IOS XE Trustpoint kan ook één Identity (ID)-certificaat importeren dat is geladen via de opdracht **voor** het **importeren** van **crypto-pki**. Het ID-certificaat is dit apparaatcertificaat dat gewoonlijk aan een service- of functie is gekoppeld.
3. Een beheerder kan de opdracht **Verifiëren** en **importeren** gebruiken op hetzelfde trustpoint (dat een ID-certificaat moet importeren dat later wordt besproken). Bij het gebruik van de autorisatie/import workflow zal het trustpoint twee certificaten bevatten (root/tussenliggend + identiteits certificaat).
4. Wanneer trustpoints worden gebruikt voor het opslaan van vertrouwde peer root/tussenliggende CA-certificaten, wordt alleen de **crypto pki-verificatie** bevel wordt vereist. In dit scenario zal een trustpoint slechts het enige certificaat bevatten dat door de beheerder wordt geverifieerd.

Opmerking: de volgende secties voor **crypto pki authenticeren** en **crypto pki import** en latere secties met voorbeelden van auth/import voor multilevel certificaten zullen verdere context geven aan deze vier kogels.

Trustpoints kunnen verschillende opdrachten hebben geconfigureerd. Deze opdrachten kunnen worden gebruikt om de waarden in een certificaatondertekeningsaanvraag (CSR) te beïnvloeden die door het apparaat is gemaakt met behulp van de opdracht **crypto pki enroll** op een trustpoint.

Er zijn veel verschillende opdrachten beschikbaar voor een trustpoint (veel te veel om in dit document in detail te treden), maar veel meer gebruikelijke voorbeelden worden gedetailleerd in het voorbeeld trustpoint en de onderstaande tabel:

```
crypto pki trustpoint labTrustpoint
 enrollment terminal pem
 serial-number none
 fqdn none
 ip-address none
 subject-name cn=router.example.cisco.com
 subject-alt-name myrouter.example.cisco.com
 revocation-check none
 rsakeypair rsaKey
 hash sha256
```

Opdracht	**Beschrijving**
crypto pki trustpoint labTrustpoint	Human readable configuratie label voor dit trustpoint. Gebruikt om te koppelen naar functies of services in latere opdrachten.
inschrijving terminal pem	Bepaalt welke actie de opdracht **crypto penroll** zal ondernemen. In dit voorbeeld geeft **de terminal pem van de inschrijving** aan dat het certificaat ondertekeningsverzoek (CSR) uitgevoerd zal worden naar de terminal in een Base64 PEM geformatteerde tekst.

	Andere opties zoals **zelf ondertekende inschrijving** kunnen worden gebruikt om een zelf-ondertekende certificaat of **inschrijvingsURL** te creëren kan worden geconfigureerd om een HTTP URL te definiëren en gebruik te maken van het Simple Certificate Enrollment Protocol (SCEP) protocol. Beide methoden vallen buiten het toepassingsgebied van dit document.
volgnummer geen	Bepaalt of de IOS XE-apparaatserie aan de CSR zal worden toegevoegd. Dit schakelt ook de prompt tijdens de crypto penroll opdracht uit.
FQDN geen	Hiermee wordt bepaald of de volledig gekwalificeerde domeinnaam (FQDN) zal worden toegevoegd aan de CSR. Dit schakelt ook de prompt tijdens de crypto penroll opdracht uit.
IP-adres geen	Hiermee wordt bepaald of het IP-adres van de IOS XE-apparaten wordt toegevoegd aan de CSR. Dit schakelt ook de prompt tijdens de crypto penroll opdracht uit.
onderwerp-naam cn=router.example.cisco.com	Geeft de X500 aan die aan de MVO wordt toegevoegd.
subject-alt-name myrouter.example.cisco.com	Vanaf IOS XE 17.9.1 kan een komma-gescheiden lijst van de waarden van de Alternatieve Naam van het Onderwerp (SAN) aan CSR worden toegevoegd.
revocation-check none	Geeft aan hoe het IOS XE-apparaat de geldigheid van het certificaat moet controleren. Opties zoals de certificaatintrekkingslijst (CRL) en het Online Certificate Status Protocol (OCSP) kunnen worden gebruikt als ze worden ondersteund door de certificeringsinstantie naar keuze. Dit wordt voornamelijk gebruikt wanneer het trustpoint wordt gebruikt door een andere geconfigureerde IOS XE-functie of -service. De status van de herroeping wordt ook gecontroleerd wanneer een certificaat met een vertrouwenspunt voor authentiek wordt verklaard.
rsakeypair rsaKey	Draagt het commando op om het RSA sleutelpaar te gebruiken met dit specifieke label. Voor ECDSA-certificaten wordt gebruik gemaakt van de opdracht "eckeypair ecKey", die verwijst naar het label van de EC-sleutel
hash sha256	Deze opdracht beïnvloedt het te gebruiken type hashingalgoritme. De opties zijn SHA1, SHA256, SHA384, SHA512

crypto pki-bestand

De opdracht **crypto pki enroll** wordt gebruikt om de inschrijvingsopdracht op een gegeven trustpoint te activeren. (Opdrachtsyntaxis)
Voor het voorbeeld trustpoint eerder weergegeven de opdracht **crypto pki enroll labTrustpoint** zal het certificaat ondertekeningsverzoek (CSR) tonen aan de terminal in Base64 PEM tekstformaat zoals in het voorbeeld hieronder.

Dit certificaat ondertekeningsverzoek kan nu in een tekstbestand of kopie worden opgeslagen en vanaf de opdrachtregel worden geplakt om aan een derde CA te kunnen leveren voor validering en ondertekening.

<#root>

```
Router(config)#
```

```
crypto pki enroll labTrustpoint
```

```
% Start certificate enrollment ..

% The subject name in the certificate will include: cn=router.example.cisco.com
% The fully-qualified domain name will not be included in the certificate
Display Certificate Request to terminal? [yes/no]:
```

```
yes
```

```
Certificate Request follows:

-----BEGIN CERTIFICATE REQUEST-----
MIICrTCCAZUCAQAwIzEhMB8GA1UEAxMYcm91dGVyLmV4YW1wbGUuY2lzY28uY29t
[..truncated..]
mGvBGUpn+cDIdFcNVzn8LQk=
-----END CERTIFICATE REQUEST-----

---End - This line not part of the certificate request---
```

crypto pki-verificatie

De **crypto pki authenticate** opdracht wordt gebruikt om een betrouwbaar CA certificaat toe te voegen aan een gegeven trustpoint. Elk trustpoint kan in één keer worden geverifieerd. Dat wil zeggen dat een trustpoint slechts één CA-root of tussenliggend certificaat kan bevatten. De opdracht een tweede keer uitvoeren en een nieuwe cert toevoegen zal het eerste certificaat overschrijven.

Met de opdrachtinschrijving **terminal** geconfigureerd zal de **crypto pki authenticate** opdracht de router voor een Base64 PEM geformatteerd certificaat te uploaden via de CLI. (Opdrachtsyntaxis)

Een beheerder kan een trustpoint authenticeren om de root en optionele tussenliggende certificaten toe te voegen in een certificaatketen met het oog op het importeren van een apparaatcertificaat later.

Een beheerder kan ook een trustpoint authenticeren om andere vertrouwde root-CA's aan het IOS XE-apparaat toe te voegen om vertrouwensrelaties met peer-apparaten mogelijk te maken tijdens protocol handshakes met dat peer-apparaat.

Om verder te illustreren, kan een peer apparaat een certificaatketting kenmerken die door "Root CA 1" wordt ondertekend. Opdat de certificaatbevestiging tijdens de protocolhanddruk tussen het IOS XE-apparaat en het peer-apparaat succesvol is; een beheerder kan **crypto pki authenticate** opdracht gebruiken om het

CA-certificaat toe te voegen aan een trustpoint op het IOS XE-apparaat.

Het belangrijkste punt om te onthouden: Het voor authentiek verklaren van vertrouwenspunten die crypto pki gebruiken verklaart is altijd voor het toevoegen van de wortel of de middencertificaten van CA aan een vertrouwenspunt; niet voor het toevoegen van identiteitscertificaten. Merk op dat dit concept ook wordt toegepast op het verifiëren van zelf-ondertekende certificaten van een ander peer apparaat.

In het onderstaande voorbeeld wordt getoond hoe u een trustpoint kunt authenticeren door eerder de opdracht **crypto pki authenticate te** gebruiken:

<#root>

```
Router(config)#
```

```
crypto pki authenticate labTrustpoint
```

```
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
[..truncated..]
-----END CERTIFICATE-----

Certificate has the following attributes:
       Fingerprint MD5: C955FC74 7AABC184 D8A75DE7 3C9E7218
      Fingerprint SHA1: 3A99FF61 1E9E6C7B D0E567A9 96D882F5 2279C534

% Do you accept this certificate? [yes/no]:
```

```
yes
```

```
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

crypto pki import

Deze opdracht wordt gebruikt om het identiteits (ID)-certificaat te importeren in een trustpoint. Eén enkel trustpoint kan slechts één enkel ID-certificaat bevatten en bij een tweede afgifte van de opdracht wordt gevraagd het eerder geïmporteerde certificaat te overschrijven. (Opdrachtsyntaxis)

In het onderstaande voorbeeld wordt getoond hoe u een Identity Certificate kunt importeren in het voorbeeld trustpoint van een eerdere toepassing met de opdracht **crypto pki import**.

<#root>

```
Router(config)#
```

```
crypto pki import labTrustpoint certificate
```

```
Enter the base 64 encoded certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
[..truncated..]
-----END CERTIFICATE-----

% Router Certificate successfully imported
```

Een beheerder krijgt een fout als het proberen om een certificaat te importeren voordat het trustpoint het CA-certificaat heeft geverifieerd dat gebruikt wordt om dit certificaat direct te ondertekenen.

<#root>

```
Router(config)#
```

```
crypto pki import labTrustpoint certificate
```

```
% You must authenticate the Certificate Authority before
 you can import the router's certificate.
```

Verificatie van peer-CA-certificaten

Peer CA-certificaten worden toegevoegd aan IOS XE met behulp van dezelfde methode om CA-certificaten toe te voegen. Dat wil zeggen, ze zijn geauthenticeerd tegen een trustpoint met behulp van de **crypto pki authenticate** opdracht.

De onderstaande opdracht toont hoe u een trustpoint kunt maken en een CA-certificaat van een peer-derde partij kunt verifiëren.

1. Maak eerst een trustpoint met een beschrijvende naam die het peer CA certificaat zal houden
2. vorm **inschrijvingsterminal pem** zodat de crypto pki authenticate opdracht vraagt om het certificaat via de opdrachtregel.
3. Configuratie van **herroeping-controle geen** om CRL/OCSP-controle tijdens het importproces over te slaan
4. Verifieer het trustpoint en specificeer het certificaat
5. Herhaal stap 1-4 voor zoals vereist voor peer CA-certificaten (vergeet niet slechts één CA-certificaat per trustpoint!)

<#root>

```
Router(config)#
```

```
crypto pki trustpoint PEER-ROOT
```

```
Router(ca-trustpoint)#
```

```
enrollment terminal pem
```

```
Router(ca-trustpoint)#
```

```
revocation-check none
```

```
Router(ca-trustpoint)#
```

```
crypto pki authenticate PEER-ROOT
```

```
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
[..truncated..]
```

```
-----END CERTIFICATE-----

Certificate has the following attributes:
       Fingerprint MD5: 62D1381E 3E03D06A 912BAC4D 247EEF17
      Fingerprint SHA1: 3C97CBB4 491FC8D6 3D12B489 0C285481 64198EDB

% Do you accept this certificate? [yes/no]:
```

yes

```
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

Verificatie van een of meer tussentijdse certificaten

In de vorige voorbeelden is beschreven hoe u een CSR kunt genereren met behulp van **crypto pki-inschrijving**, hoe u het basiscertificaat van een CA-certificaat verifieert met behulp van **crypto pki-authenticatie** en vervolgens het identiteitscertificaat importeert met behulp van **crypto pki-import**. Bij de invoering van tussentijdse certificaten verschilt het proces echter enigszins. Angst niet, dezelfde concepten en commando's zijn nog steeds van toepassing! Het verschil zit hem in de manier waarop de trustpunten die de certificaten bezitten, zijn vastgelegd.

Herinner dat elk trustpoint slechts één enkel Root of MiddenCA certificaat kan bevatten. In een voorbeeld waar we een CA-keten hebben zoals onder de hieronder weergegeven, is het onmogelijk om de crypto pki authenticate opdracht te gebruiken om meer dan één CA-certificaat toe te voegen:

<#root>

```
- Root CA


 - Intermediate CA 1


   - Identity Certificate
```

Oplossing:

1. Maak een trustpoint dat de geverifieerde root-CA bevat.
2. Verifieer vervolgens het tussentijds certificaat met het trustpoint dat wordt gebruikt om de CSR te maken
3. Ten slotte moet het identiteitsbewijs worden ingevoerd in het uiteindelijke vertrouwenspunt.

Met behulp van de onderstaande tabel kan men het certificaat te bevelen aan trustpoint mapping met kleuren die corresponderen met de vorige keten om te helpen met visualisatie.

Naam certificaat	Vertrouwpunt voor gebruik	Opdracht voor gebruik
Root CA	crypto pki trustpoint **ROOT-CA**	crypto API authenticeren **ROOT-CA**
Tussenfase-CA 1	crypto pki trustpoint **labTrustpoint**	crypto-sleutel authenticeren **labTrustpoint**

Identiteitscertificaat	crypto pki trustpoint **labTrustpoint**	crypto point import **labTrustpoint** certificate

Dezelfde logica kan worden toegepast op een certificaatketen met twee tussenliggende CA-certificaten. Opnieuw worden de kleuren verstrekt om met de visualisering van te helpen waar het nieuwe Intermediate CA op de IOS XE configuratie wordt toegepast.

<#root>

- Root CA

- Intermediate CA 1

- Intermediate CA 2

- Identity Certificate

Naam certificaat	**Vertrouwpunt voor gebruik**	**Opdracht voor gebruik**
Root CA	crypto pki trustpoint **ROOT-CA**	crypto API authenticeren **ROOT-CA**
Tussenfase-CA 1	crypto pki trustpoint **INTER-CA**	crypto pki authenticeren **INTER-CA**
Tussenfase-CA 2	crypto pki trustpoint **labTrustpoint**	crypto-sleutel authenticeren **labTrustpoint**
Identiteitscertificaat	crypto pki trustpoint **labTrustpoint**	crypto point import **labTrustpoint** certificate

Als je goed kijkt, zie je twee patronen:

1. Alle Root- of Intermediate-certificaten worden geladen in trustpoints met behulp van **crypto pki authenticate** (ongeacht hoeveel er zijn).
2. Je kunt ook opmerken dat het definitieve certificaat vóór het identiteitsbewijs van het apparaat (lees het certificaat dat het identiteitsbewijs direct ondertekende) altijd wordt geauthentiseerd op hetzelfde vertrouwenspunt waar het identiteitsbewijs moet worden geïmporteerd.
    - Gelijkaardig aan de vroeger getoonde fout, zal IOS XE geen beheerder een certificaat laten invoeren zonder het certificaat van CA eerst voor authentiek te verklaren dat wordt gebruikt om dit certificaat direct te ondertekenen.

Deze twee patronen hierboven kunnen worden gebruikt voor elk aantal tussenliggende certificaten na twee, hoewel in de meeste implementaties een beheerder waarschijnlijk meer dan twee tussenliggende CAâ€™s in een certificaatketen zal zien.

Voor de volledigheid wordt ook de volgende Root/Identity Certificate-tabel verstrekt:

<#root>

- Root CA

- Identity Certificate

Naam certificaat	Vertrouwpunt voor gebruik	Opdracht voor gebruik
Root CA	crypto pki trustpoint **labTrustpoint**	crypto-sleutel authenticeren **labTrustpoint**
Identiteitscertificaat	crypto pki trustpoint **labTrustpoint**	crypto point import **labTrustpoint** certificate

Verificatie

- Tijdens het authenticatie- of importproces worden door IOS XE verschillende echtheidscontroles uitgevoerd om er zeker van te zijn dat het certificaat geldig en goed gevormd is. Deze fouten zullen worden afgedrukt op het scherm of logs (toon vastlegging) zoeken naar lijnen die beginnen met "CRYPTO_PKI"

Hieronder worden enkele veelvoorkomende voorbeelden beschreven:

Geldige voor/na-controles worden uitgevoerd op basis van de ingestelde tijd versus de tijd die in het certificaat wordt gevonden

<#root>

004458:

Aug 9

21:05:34.403: CRYPTO_PKI: trustpoint labTrustpoint authentication status = 0

%CRYPTO_PKI: Cert not yet valid or is expired -

start date: 05:54:04 EDT

Aug 29

2019
end date: 05:54:04 EDT Aug 28 2022

als revocation-check niet uitgeschakeld is, voert IOS XE een herroepingscontrole uit via de geconfigureerde methode voordat het certificaat wordt geïmporteerd

<#root>

003375: Aug 9 20:24:14:

%PKI-3-CRL_FETCH_FAIL: CRL fetch for trustpoint ROOT failed

003376: Aug 9 20:24:14.121:

CRYPTO_PKI: enrollment url not configured

Om details over vertrouwde configuratie, geverifieerd of geïmporteerd te bekijken, gebruikt u de onderstaande opdrachten:

```
show crypto pki trustpoints trustpoint_name
show crypto pki certificates trustpoint_name
show crypto pki certificates verbose trustpoint_name
```

Probleemoplossing

Bij het debuggen van importproblemen of andere PKI-problemen gebruikt u de volgende debugs.

```
debug crypto pki messages
debug crypto pki transactions
debug crypto pki validation
debug crypto pki api
debug crypto pki callback
!
debug ssl openssl error
debug ssl openssl msg
debug ssl openssl states
debug ssl openssl ext
```

Geavanceerde IOS PKI-concepten

Een PKCS12-geformatteerd certificaat importeren

Sommige CA-providers kunnen bestanden teruggeven in PKCS#12 formaat (.pfx, .p12).

PKCS#12 is een speciaal type certificaatformaat waarbij de volledige certificaatketen van basiscertificaat via identiteitscertificaat wordt gebundeld samen met het rsa-sleutelpaar.

Deze indeling is zeer handig voor importeren met IOS XE en kan eenvoudig worden geïmporteerd met de onderstaande opdracht:

<#root>

```
Router(config)#
```

crypto pki import PKCS12-TP pkcs12 terminal password Cisco123

or

```
Router(config)#
```

crypto pki import PKCS12-TP pkcs12 ftp://cisco:cisco@192.168.1.1/certificate.pfx password Cisco123

```
% Importing pkcs12...
Address or name of remote host [192.168.1.1]?
Source filename [certificate.pfx]?
Reading file from ftp://cisco@192.168.1.1/certificate.pfx!
```

```
[OK - 2389/4096 bytes]
% You already have RSA keys named PKCS12.
% If you replace them, all router certs issued using these keys
% will be removed.
% Do you really want to replace them? [yes/no]:
```

yes

```
CRYPTO_PKI: Imported PKCS12 file successfully.
```

PKCS12- of PEM-certificaten exporteren

Een beheerder kan certificaten naar de terminal exporteren als Base64 Plain Text PEM, Base64 versleutelde Plain Text, of PKCS12-formaat voor importeren in andere peer-apparaten.

Dit is handig bij het opbrengen van nieuwe peer-apparaten en een beheerder moet een Root CA-certificaat delen dat het apparaatidentiteitscertificaat heeft ondertekend.

Hieronder vindt u enige voorbeeldsyntaxis:

<#root>

```
Router(config)#
```

crypto pki export labTrustpoint pem terminal

```
Router(config)#
```

crypto pki export labTrustpoint pem terminal 3des password Cisco!123

```
Router(config)#
```

crypto pki export labTrustpoint pkcs12 terminal password cisco!123

RSA-toetsen exporteren

Het kan noodzakelijk zijn om RSA-sleutels te exporteren voor import naar een ander apparaat of voor gebruik bij het oplossen van problemen. Ervan uitgaande dat het sleutelpaar is gemaakt als exporteerbaar, kunnen de sleutels worden geëxporteerd met de opdracht voor het exporteren van crypto-sleutels, samen met een coderingsmethode (DES, 3DES, AES) en een wachtwoord.

Steekproefgebruik:

<#root>

```
Router(config)#
```

crypto key export rsa rsaKey pem terminal aes Cisco!123

```
% Key name: IOS-VG
   Usage: General Purpose Key
   Key data:
-----BEGIN PUBLIC KEY-----
[..truncated..]
-----END PUBLIC KEY-----
```

```
base64 len 1664-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,40E087AFF0886DA7C468D2084A0DECFB

[..truncated..]
-----END RSA PRIVATE KEY-----
```

Als de toets niet kan worden geëxporteerd, wordt een fout weergegeven.

<#root>

```
Router(config)#
```

crypto key export rsa kydavis.cisco.com pem terminal 3des mySecretPassword

```
% RSA keypair kydavis.cisco.com' is not exportable.
```

RSA-sleutels importeren die niet in het vak zijn gegenereerd

Sommige beheerders kunnen RSA en certificaatcreatie off-box uitvoeren, is het mogelijk om de RSA-sleutels te importeren met behulp van de opdracht **crypto key import** zoals hieronder getoond met het wachtwoord.

<#root>

```
Router(config)#
```

crypto key import rsa rsaKey general-purpose exportable terminal mySecretPassword

```
% Enter PEM-formatted public General Purpose key or certificate.
% End with a blank line or "quit" on a line by itself.
-----BEGIN PUBLIC KEY-----
[..truncated..]
-----END PUBLIC KEY-----

% Enter PEM-formatted encrypted private General Purpose key.
% End with "quit" on a line by itself.
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,9E31AAD9B7463502
[..truncated..]
-----END RSA PRIVATE KEY-----
quit
% Key pair import succeeded.
```

RSA-toetsen verwijderen

Gebruik de opdracht **crypto key zeroize rsaKey** om een RSA keypair genaamd rsaKey te verwijderen.

Cisco Trusted CA-bundel via Trustpool importeren

Trustpools variëren enigszins van een trustpoint, maar het kerngebruik is hetzelfde. Waar trustpoints gewoonlijk één CA-certificaat bevatten, zal een trustpool een aantal vertrouwde CAâ€™s bevatten.

Cisco publiceert CA-bundels op https://www.cisco.com/security/pki/

Een algemeen gebruik is om het ios_core.p7b bestand te downloaden met behulp van de onderstaande opdracht:

<#root>

```
Router(config)#
```

crypto pki trustpool import clean url http://www.cisco.com/security/pki/trs/ios_core.p7b

```
Reading file from http://www.cisco.com/security/pki/trs/ios_core.p7b
Loading http://www.cisco.com/security/pki/trs/ios_core.p7b
% PEM files import succeeded.
Router(config)#
```

Veelgestelde vragen

Maakt het schrappen van een trustpoint MVO of een certificaatketen die door een bepaalde MVO is toegekend ongeldig?

Nee, zodra de MVO is gegenereerd en opgeslagen kan het trustpoint worden verwijderd en opnieuw toegevoegd zonder de MVO ongeldig te maken.

Dit wordt vaak gebruikt door de technische ondersteuning van Cisco om vers te beginnen wanneer het verifiëren/importeren van certificaten fout is gegaan.

Zolang de beheerder of support engineer RSA-sleutels niet regenereren, kan de CSR of ondertekende certificaatketen geïmporteerd worden.

Belangrijk! Het verwijderen van de trustpoint **ZAL** alle geauthenticeerde / geïmporteerde certificaten die problematischer zouden kunnen zijn als die certificaten momenteel in gebruik zijn door een of andere dienst of functie.

Zal het genereren van een MVO op een trustpoint het bestaande certificaat ongeldig maken?

Nee, dat is gebruikelijk wanneer certificaten bijna verlopen zijn. Een beheerder kan een opdracht **crypto pki enroll** uitvoeren om een nieuwe CSR te maken en het proces voor het ondertekenen van certificaten te starten met een CA, terwijl de bestaande certificaten die zijn geverifieerd/geïmporteerd, in gebruik blijven. Het moment waarop een beheerder de certificaten vervangt door **crypto pki authenticate**/**crypto pki import** is het moment waarop de oude certificaten worden vervangen.

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.